FaIMS: A fast algorithm for the inverse medium problem with multiple frequencies and multiple sources for the scalar Helmholtz equation

S. Chaillat G. Biros

Abstract

We propose an algorithm to compute an approximate singular value decomposition (SVD) of least-squares operators related to linearized inverse medium problems with multiple events. Such factorizations can be used to accelerate matrix-vector multiplications and to precondition iterative solvers.

We describe the algorithm in the context of an inverse scattering problem for the low-frequency time-harmonic wave equation with broadband and multi-point illumination. This model finds many applications in science and engineering (e.g., seismic imaging, subsurface imaging, impedance tomography, non-destructive evaluation, and diffuse optical tomography).

We consider small perturbations of the background medium and, by invoking the Born approximation, we obtain a linear least-squares problem. The scheme we describe in this paper constructs an approximate SVD of the Born operator (the operator in the linearized least-squares problem). The main feature of the method is that it can accelerate the application of the Born operator to a vector.

If N_ω is the number of illumination frequencies, N_s the number of illumination locations, N_d the number of detectors, and N the discretization size of the medium perturbation, a dense singular value decomposition of the Born operator requires $\mathcal{O}(\min(N_s N_\omega N_d, N)]^2 \times \max(N_s N_\omega N_d, N))$ operations. The application of the Born operator to a vector requires $\mathcal{O}(N_\omega N_s \mu(N))$ work, where $\mu(N)$ is the cost of solving a forward scattering problem. We propose an approximate SVD method that, under certain conditions, reduces these work estimates significantly. For example, the asymptotic cost of factorizing and applying the Born operator becomes $\mathcal{O}(\mu(N) N_\omega)$. We provide numerical results that demonstrate the scalability of the method.

1 Introduction

Let $\mathcal{B}(u, \eta)$ be a bilinear operator in which η is an the medium perturbation and u is the state (total scattered field). Given η, $\mathcal{B}$ is a well-posed boundary value problem for u. We consider the following problem: given data $\phi_i = \mathcal{D}u_i$ we want to reconstruct η. Here u_i is the solution of $\mathcal{B}(u_i, \eta) = f_i$, $i = 1, \ldots, N_e$, $\mathcal{D}$ is a linear observation operator, f_i is a known illumination source, and N_e is the number of illumination events. In this paper, we introduce an approximate factorization algorithm for the linearization of the operator that maps η to ϕ_i. Such a factorization can be used with an iterative inversion scheme [9] or with Bayesian estimation algorithms to estimate covariances [14, 32].

Inverse medium problems find applications in elasticity (full waveform inversion seismology), electromagnetics (ground penetrating radar), acoustics (non-destructive evaluation), diffusive transport (optical tomography), electrostatics (impedance tomography), and other applications. To make ideas concrete, we consider the free-space frequency-domain acoustic scattering. We describe a numerical algorithm for the Born approximation formulation of the inverse medium problem in scalar scattering [10]. Given $N_e = N_s N_\omega$ incident ("illumination") fields $\{u(\mathbf{r}; s, \omega)\}_{s=1,\omega=1}^{N_s,N_\omega}$ (where s indexes the spacial location of the source of the

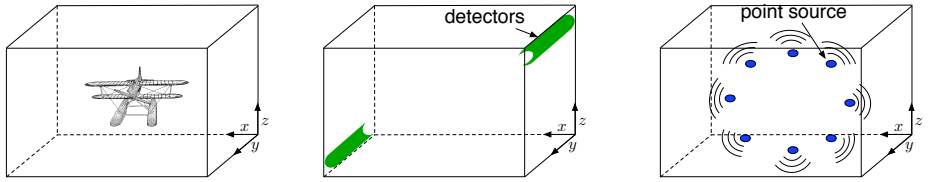
incident field and ω indexes its frequency), we record the scattered field $\boldsymbol{\phi}(\mathbf{r}_d; s, \omega)$ at N_d detector locations $\{\mathbf{r}_d\}_{d=1}^{N_d}$ and we seek to recover the medium perturbation $\eta(\mathbf{r})$ by solving

$$\phi(\mathbf{r}_d; s, \omega) = \int_H G(\mathbf{r}_d, \mathbf{r}; \omega)\eta(\mathbf{r})u(\mathbf{r}; s, \omega)\, d\mathbf{r} \tag{1}$$

for η. This is a Lippmann-Schwinger scattering equation, where $G(\cdot, \cdot; \omega)$ is the Green's function for the given scattering problem (in general not known analytically but computable by solving a boundary value problem) at frequency ω, H is the support of η (assumed to be known), and $\mathbf{r}$ is a point in H. Upon discretization using N quadrature points, we have

$$\phi(\mathbf{r}_d; s, \omega) = \sum_{j=1}^{N} G(\mathbf{r}_d, \mathbf{r}_j; \omega)\eta(\mathbf{r}_j)u(\mathbf{r}_j; s, \omega), \tag{2}$$

where the quadrature weights have been absorbed in $\eta(\mathbf{r}_j)$ and by using "=" we ignore the quadrature discretization error. Indeed, in the rest of the paper (2) will be considered exact, that is, ϕ will be generated by point scatterers located at the quadrature points $\{\mathbf{r}_j\}_{j=1}^{N}$ with scattering strengths $\{\eta(\mathbf{r}_j)\}_{j=1}^{N}$. The problem is summarized in Figure 1.



Complex unknown object | Detectors placed on arbitrary geometries | Spherical wave sources placed at arbitrary locations

Figure 1: *We propose an algorithm for the Born approximation of the inverse medium problem. For simplicity, we assume that the medium perturbation is represented by a set of point scatterers in a 3-D domain H. The data consists of measurements of the scattered field generated by separate incident fields. In our experiments, the incident fields are generated by point sources that illuminate the region of interest—possibly at multiple frequencies. Both sources and detectors can be located in arbitrary positions as opposed to analytic techniques which require regular geometries.*

Equation (2) is linear on η. We introduce $\mathbf{M}$ to denote the mapping from η to $\boldsymbol{\phi}$, so that

$$\mathbf{M}\boldsymbol{\eta} = \boldsymbol{\phi}.$$

Note that if we write the normal equations for this system, we obtain $\mathbf{M}^*\mathbf{M}\boldsymbol{\eta} = \mathbf{M}^*\boldsymbol{\phi}$; the operator $\mathbf{M}^*\mathbf{M}$ in the normal equations is also known as the Hessian (or reduced Hessian depending on the formulation.)

Depending on the maximum illumination frequency, and the distance between H the detectors and the sources, $\mathbf{M}$ may have a numerically low rank. We will like to compute an approximation to $\mathbf{M}$ so that the cost of applying $\mathbf{M}$ or its pseudoinverse to a vector are as small as possible. For example, when reconstructing η one approach is to use a Krylov iterative method like the LSQR and Conjugate Gradient (CG) method for the normal equations [9, 29]. To analyze the cost of such an iterative solver, let us define the cost of the forward scattering solver as $\mu(N, N_d)$, or simply $\mu(N)$ (assuming $N_d \leq N$). Then, the cost of an iterative method would be $N_s N_\omega \mu(N)$ per matvec, which can be quite high if $N_s N_\omega$ is large or if we

have an expensive forward scattering solver. Also using a Krylov method requires preconditioning typically based on limited-memory BFGS or Lanczos preconditioners [2, 16, 20]. Constructing BFGS preconditioners has similar complexity with inverting $\mathbf{M}$ [27]. Lanczos methods can be used and offer a viable alternative to our approach but are more difficult to implement correctly [18]. Another approach is to use spectral preconditioning methods [1] or multigrid [6, 3]. Our method can be combined with spectral preconditioning methods, for example construct coarse and fine spaces.

Finally one could form $\mathbf{M}$ and use a dense factorization algorithm, say, use an classical SVD factorization [15]. A dense SVD is prohibitively expensive[1] because its work complexity is $\mathcal{O}(\min(N_s N_\omega N_d, N)]^2 \times \max(N_s N_\omega N_d, N))$.

Contributions. To summarize, our goal is to design an algorithm that approximates $\mathbf{M}$ and scales "well" with N, N_ω, N_s and N_d, for the low frequency regime.[2] Our main contribution is the construction of $\mathbf{M}_\varepsilon$, an approximate singular value decomposition for $\mathbf{M}$ based on the following algorithmic components:

- a rank-revealing randomized factorization. We have used the algorithm 4.4 proposed in [18] that combines randomization with the power iteration.
- preprocessing of the incident field u using an SVD to transform the incoming field and data and reduce the dimension of N_s;[3]
- and a recursive SVD that can be used to approximately compute the SVD of $\mathbf{M} = [\mathbf{M}_1\ \mathbf{M}_2]^t$ given the approximate SVDs of $\mathbf{M}_1$ and $\mathbf{M}_2$.

Using these components, we construct $\mathbf{M}_\varepsilon$ an approximate SVD factorization for $\mathbf{M}$ whose, given the incident field, the total work complexity is $\mathcal{O}(N_\omega \mu(N, N_d) R_s)$, where R_s depends on the problem geometry and the maximum frequency but is independent of N_s, asymptotically. In our implementation, we use a direct evaluation for the scattered field $\mu(N, N_d) = N N_d$. Using a fast multipole acceleration the complexity can be reduced to $\mu(N, N_d) = N + N_d$ for the low frequency. We test our algorithm on problems in which the scatterer size varies from $1/10$ to five wavelengths. Our algorithm supports arbitrary distributions of sources, detectors and frequencies. For the purposes of demonstrating the quality of the approximation, we use one of the simplest methods of solving inverse problems, the truncated SVD. [4]

Outline of FaIMS: First, we reduce the number of incident fields from N_s to R_s using the randomized SVD. Then, we decompose $\mathbf{M}$ into N_ω smaller submatrices $\mathbf{M}^\omega$ of size $R_s N_d \times N$ ($1 \le \omega \le N_\omega$). We compute the approximate SVD of each small matrix by using the randomized SVD. We apply a low rank approximation whenever possible, leading to a compression of the matrix and a speed-up of the computations. We combine the approximate SVDs of the $\mathbf{M}^\omega$ to approximate the SVD of $\mathbf{M}$, $\mathbf{M}_\varepsilon$, using the recursive SVD. This recursive SVD provides a precise characterization of the inverse problem and allows us to easily apply the pseudo-inverse of $\mathbf{M}_\varepsilon$ to the data. We have termed the overall algorithm "*FaIMS*". This algorithm

[1] For example, if $N_\omega = 10$, $N_s = 100$, $N_d = 10^2$, and $N = 100^3$, we will need over one month of computation to compute the SVD on a single core 2 Gigaflops/sec machine.

[2] We mean that given a minimum and a maximum frequency, the algorithm scales well as we increase the number of sample frequencies within the given frequency range.

[3] This preprocessing step, as we describe it in this paper is valid only in the case in which the detectors are the same for all of the sources.

[4] To clarify, we do not advocate using the truncated SVD as an inversion method and do not we use it to demonstrate a new inversion scheme since we do not consider noise and regularization issues.

can handle efficiently a large number of sources and frequencies which lead to better resolution (Fig. 2). The storage complexity of FaIMS is $\mathcal{O}(NR)$, where R is the overall rank of the approximation to $\mathbf{M}$. FaIMS achieves this complexity estimate since it does not require the assembly of $\mathbf{M}$. It only requires matrix-vector multiplications with submatrices of $\mathbf{M}$.

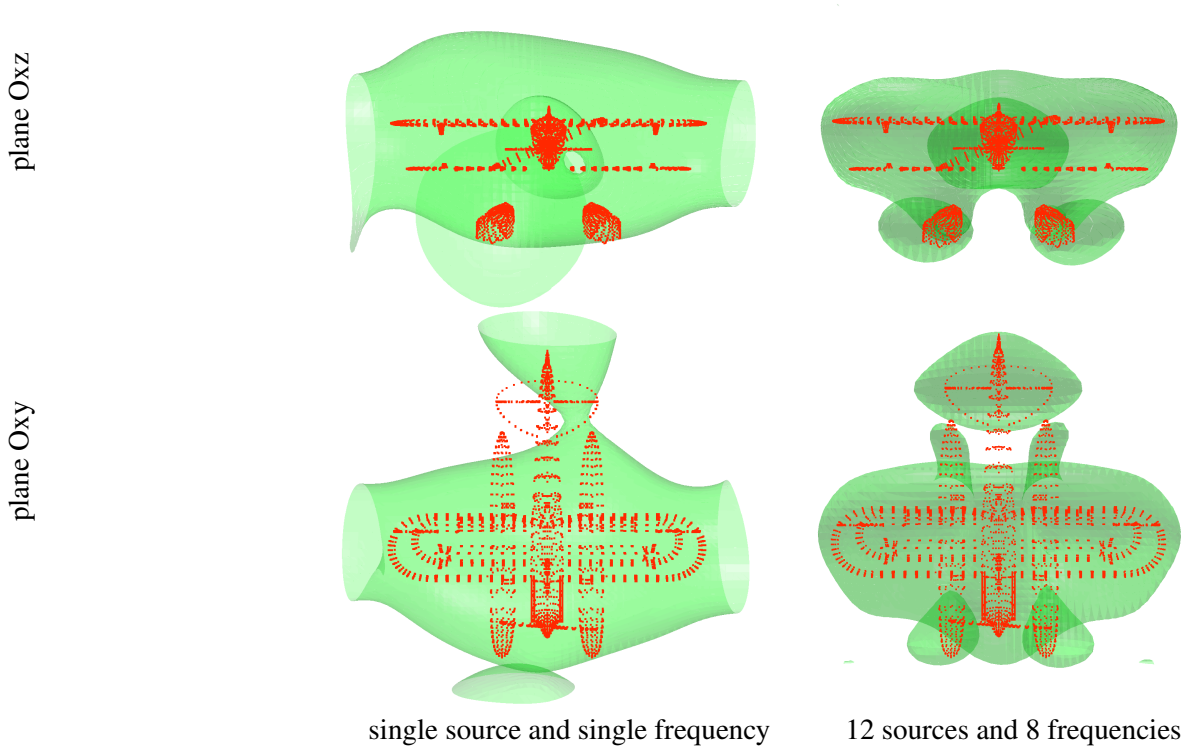


Figure 2: *We report the isosurfaces $\boldsymbol{\eta} = 0.25\boldsymbol{\eta}_{max}$ for the scatterer model of a biplane with size one wavelength. The incident field for results on the left column is generated with one point sources excited at a single frequency. For the results of the right columns, we generate the incident field by using* 12 *sources and* 8 *frequencies. The mesh size is* $N = [51]^3$ *and the scattered field is measured at* 162 *detectors located on a sphere enclosing the biplane. The results of the inversion with a single source and single frequency already enables to find the object location but the addition of more sources and frequencies permits to obtain better accuracy.*

Limitations. FaIMS, works well when (1) the detectors, the sources, and the support of η are well separated and (2) the maximum excitation frequency is small enough. If these two conditions are met, the mapping $\boldsymbol{\phi} = \mathbf{M}\boldsymbol{\eta}$ will be sufficiently low rank to result in computational savings. If the first condition (well-separateness) is not met, a more elaborate block decomposition of $\mathbf{M}$ is required (resembling tree decompositions for fast-multipole methods and hierarchical matrices) to construct approximations to $\mathbf{M}$. The second condition (maximum frequency) is much harder to circumvent.

Our analysis in this paper is not related to a specific inversion scheme but on approximating $\mathbf{M}$ in order to accelerate its application to a vector. To demonstrate the approximation, we use truncated SVD for synthetic examples in which the data has no noise. In a realistic inverse problem one needs to account for noise, model errors, approximation errors, and incorporate more sophisticated regularization techniques.

Also, we are considering neither sparse reconstruction ideas for η [7, 19] nor adaptive reconstruction schemes [4, 17]. We assume that the location of the detectors is independent of the source location and frequency.

In our computations, we know the Green's function in analytic form so that the scattered field due to N scatterers can be evaluated at N_d detectors in $\mathcal{O}(N + N_d)$ work and storage using a fast multipole scheme [8, 33]. However, this is not a fundamental limitation of the algorithm. Any forward-scattering method with good complexity and accuracy features can be used in FaIMS without changing the behavior of the algorithm. In higher frequencies, such solvers are harder to construct. In the general case one needs to solve a forward problem in order to evaluate the scattered field given a background medium.

Finally, we haven't pursued randomization in the frequency domain. The input-output operator depends nonlinearly on the frequency and randomization techniques for linear operators are not directly applicable. However, the block decomposition of FaIMS allows a heuristic greedy algorithm for the different frequencies. We will present results on this approach in a future paper.

Related work for problems with multiple sources. Our work has been inspired by the work in [25], in which a fast analytic SVD based on Fourier analysis was used for the case in which the sources and detectors are uniformly distributed on the boundary of a regular geometry (plane, cylinder, or sphere) and the scatterer is uniformly discretized in the domain of the corresponding regular geometry. The problems considered in [25] were reconstructions of absorption and diffusion coefficients for optical tomography problems formulated in the frequency domain. With FaIMS, we can consider detectors on arbitrary geometries and point sources at arbitrary locations, as FaIMS only requires a fast forward scattering solver.

Let us mention that there is work for forward and inverse problems with multiple sources in the geophysics community for reducing the number of sources using linear combinations [22, 28, 19]. However, we could not find work that extends of these ideas to the inverse medium problem.

Our method can be viewed as an operator approximation. In general, there exist several methods to approximate and compress matrices based on randomization. The method we are using here was developed in a series of papers in [23, 26]. Other approximation schemes (for example the CUR matrix decomposition developed in [24]) could be used, however the algorithm described in [18] requires only matrix-vector multiplications, it is easy to implement and has nearly optimal accuracy and stability properties when combined with the power method.

Our recursive factorization scheme essentially uses the randomized SVD on blocks of $\mathbf{M}$ and combines them again using the randomized SVD. Alternative block recursive QR factorizations [11, 12] can be used instead. The latter approach is more accurate than our scheme (see §4.2). The main advantage of our method is that it is much simpler to implement.

Finally, let us comment on randomized SVD-like decompositions for high-order tensors [21]. Such decompositions are relevant because the forward operator $\mathbf{M}$ can be viewed as a third-order tensor that maps the incident field and the medium perturbation to data. One could explore a randomized tensor decomposition, but we have not pursued this approach in this paper.

Outline. In Section 2, we state the problem formulation. In Section 3, we give a summary of the algorithm FaIMS. In Section 4 we present the SVD algorithms that are required in the overall method (presented in Section 5). Finally, in Section 6, we present numerical results for the reconstruction of various point scatterer locations.

Notation. In Table 1, we summarize all the symbols used throughout this article. We use Roman letters to denote continuous scalar fields and operators, bold lower case letters to denote finite dimensional vectors, and bold upper case letters to denote finite dimensional linear operators.

$k(\mathbf{r})$	wavenumber;
k_0	background medium wavenumber;
λ	wavelength $\lambda = 2\pi/k$;
$\eta(\mathbf{r})$	perturbation of the background medium; $k^2(\mathbf{r}) = k_0^2 + \eta(\mathbf{r})$;
G	Green's function of the homogeneous infinite medium characterized by k_0;
H	support of the anomaly η (a cube of L^3);
L	edge size of H;
ω	indexes the frequency of the incident field;
s	indexes the location of the source of the incident field;
N	number of point scatterers in H;
N_d	number of detectors;
N_ω	number of incident wave frequencies $(\omega_1, \ldots, \omega_{N_\omega})$;
N_s	number of spherical wave source locations (point sources);
$\mathbf{M}$	overall input-output operator ($\in \mathbb{C}^{N_s N_\omega N_d \times N}$);
$\mathbf{r}$	position in space z);
$\mathbf{r}_d$	detector locations;

Table 1: *List of the main symbols used in this article.*

2 Definition of the inversion formula

The time-harmonic scalar wave equation is given by

$$\nabla^2 v(\mathbf{r}) + k^2(\mathbf{r}) v(\mathbf{r}) = -s(\mathbf{r}), \tag{3}$$

where s is the source term and k is the wavenumber. We consider the case $k^2(\mathbf{r}) = k_0^2 + \eta(\mathbf{r})$, where k_0^2 is the wavenumber of the background medium and η is the unknown perturbation. If we denote the total scattered field as the sum of the incident field $\mathbf{u}$ and the scattered field $\boldsymbol{\phi}$, $\mathbf{v} = \boldsymbol{\phi} + \mathbf{u}$, eq. (3) becomes

$$\nabla^2 \phi(\mathbf{r}) + k_0^2 \phi(\mathbf{r}) = -\eta(\mathbf{r})(\phi(\mathbf{r}) + u(\mathbf{r})). \tag{4}$$

Using the Born approximation, we neglect $-\eta(\mathbf{r})\phi(\mathbf{r})$ and we obtain

$$\nabla^2 \phi(\mathbf{r}) + k_0^2 \phi(\mathbf{r}) = -\eta(\mathbf{r}) u(\mathbf{r}). \tag{5}$$

We introduce the free-space Green's function G given by

$$G(\mathbf{r}, \mathbf{r}') = \frac{\exp(ik_0|\mathbf{r} - \mathbf{r}'|)}{4\pi|\mathbf{r} - \mathbf{r}'|}. \tag{6}$$

The solution of eq. (5) can be obtained as a convolution with G and is given by

$$\phi(\mathbf{r}_d; s, \omega) = \int_H G(\mathbf{r}_d, \mathbf{r}; \omega) \eta(\mathbf{r}) u(\mathbf{r}; s, \omega) d\mathbf{r}. \tag{7}$$

Equation (7) is the forward problem, in which, given η, we can compute $\boldsymbol{\phi}$. In the inverse problem, we seek to recover the anomaly η given $\phi(\mathbf{r}_d; s, \omega)$, a set of measurements generated by $N_s N_\omega$ known incident fields and measured at N_d detector locations.

After discretization of $\eta(\mathbf{r})$ (which we will denote it by $\boldsymbol{\eta} \in \mathbb{C}^N$), for a given source s, frequency ω and detector d (the quadrature weights are absorbed in $\boldsymbol{\eta}$ and we ignore the discretization error) equation (7) becomes

$$\boldsymbol{\phi}^\omega_{ds} = \sum_j^N \mathbf{G}^\omega_{dj} \mathbf{u}^\omega_{js} \boldsymbol{\eta}_j,$$

which we write in a matrix form $\boldsymbol{\phi} = \mathbf{M}\boldsymbol{\eta}$. To demonstrate the quality of our approximation to $\mathbf{M}$, we invert for $\boldsymbol{\eta}$ by first computing $\mathbf{M}_\varepsilon$, the approximate SVD of $\mathbf{M}$, which we then use to apply the pseudo-inverse of $\mathbf{M}_\varepsilon$ on $\boldsymbol{\phi}$. In our experiments the incident field is $u(\mathbf{r}; s, \omega) = G(\mathbf{r}; s, \omega)$, a spherical wave corresponding to a point source.

3 Summary of FaIMS

Before presenting the details of FaIMS (section 5) let us outline the basic steps in the algorithm. Recall that our main goal is to avoid the $N_s N_\omega N_d N$ complexity of applying $\mathbf{M}$ to a vector.

We introduce a preprocessing step in which use singular value decomposition to reduce the number of incident fields. This step is analogous to source recombination techniques that have appeared in the literature. In the next two steps, we compute the inputs for the recursive SVD (section 4.2): the approximate SVD of $\mathbf{M}^\omega$ for a fixed frequency ω. Then we use the recursive SVD to combine the individual approximate SVDs for each frequency. Overall, FaIMS has four main steps:

1. **Incident field SVD.** For each frequency, we preprocess the incident field $\mathbf{u}^\omega$ using the randomized SVD [18] (section 4.1) to compress the number of incoming fields $\mathbf{u}^\omega$ and data $\boldsymbol{\phi}^\omega$ and reduce the number of sources from N_s to R^ω_s.

2. **Forward problem SVD.** For every frequency, we compute the approximate SVD of the Green's function $\mathbf{G}^\omega$ by applying the randomized SVD. Each matrix $\mathbf{G}^\omega \in \mathbb{C}^{N_d \times N}$ is approximated by a matrix of rank by lower or equal to a constant which we denote R^ω_g. [5]

3. **Single frequency-multiple sources SVD**. Once we have computed the approximate SVDs of the Green's functions, we can combine them for a fixed frequency using the algorithm presented in section 5. Each matrix $\mathbf{M}^\omega \in \mathbb{C}^{R^\omega_s R^\omega_g \times N}$ is approximated by a matrix of rank R_ω.

4. **Overall SVD**. Using the results of step C, we apply the recursive SVD (section 4.2) to obtain the approximate SVD of the complete system matrix $\mathbf{M}$.

As mentioned before, once $\mathbf{M}_\varepsilon$, the approximate SVD of $\mathbf{M}$ is computed, we can combine it with a regularization operator and solve the linearized medium problem using an iterative method.

In the following section, we present the randomized and recursive algorithms that we use to construct $\mathbf{M}_\varepsilon$. The randomized SVD allows fast approximation of low-rank matrices, using a matrix-vector multiplication. Then the recursive SVD enables to obtain the approximate SVD of the matrix $\mathbf{M} = [\mathbf{M}_1 \quad \mathbf{M}_2]^t$ given the approximate SVDs of $\mathbf{M}_1$ and $\mathbf{M}_2$. This algorithm is faster than a standard SVD when low rank

[5] The compression of the Green's function seems specific to our forward problem formulation; it is not. A finite element or finite-difference-based forward solver can be used in place of $\mathbf{G}$.

approximations of $\mathbf{M}_1$ and $\mathbf{M}_2$ are available. It also provides better memory locality than applying the randomized SVD directly to $\mathbf{M}$. In Table 3, we summarize the notation for the approximate ranks of the operators that appear in FaIMS.

step	A	B	C	D
approximate rank	R_s^ω	R_g^ω	R_ω	R
size initial matrix	$N_s \times N$	$N_d \times N$	$R_s^\omega R_g^\omega \times N$	$N_\omega R^\omega \times N$

Table 2: *Notation for the approximate ranks of operators that appear in the four steps of FaIMS.*

4 Randomized and recursive SVDs

4.1 Randomized SVD

There has been a significant amount of work on randomized algorithms for low rank approximations of matrices. In our work, we use an algorithm proposed in [18] (algorithm 4.4). We briefly summarize its main steps here for completeness but we omit the power iteration, which is required in order to obtain the error estimates we give below. Let $\mathbf{M}$ be a matrix of size $m \times n$. Then the randomized SVD method computes r, Φ, Λ and Ψ, with singular values $\sigma_1 \leq \sigma_2 \leq \ldots \leq \sigma_{\min(m,n)}$ such that $\|\Phi\Lambda\Psi^* - \mathbf{M}\| \leq \varepsilon\sigma_{r+1}$. Here σ_{r+1} is the $r+1$ singular value of $\mathbf{M}$ and * denotes the conjugate transpose. Here Φ and Ψ are matrices of size respectively $m \times r$ and $n \times r$, where r are the number of singular values greater than a prescribed accuracy ε. Λ is the diagonal matrix of size $r \times r$ containing the corresponding singular values.

Computing the approximate SVD requires finding $\mathbf{Q}$ such that

$$\|\mathbf{Q}\mathbf{Q}^*\mathbf{M} - \mathbf{M}\| \leq \varepsilon\sigma_{r+1}, \tag{8}$$

end the constant ε is nearly independent of the size and rank of the matrix when the power iteration is used. So for notational simplicity in the following $\varepsilon \approx= 1$. The main component of the algorithm is the application of $\mathbf{M}$ to a random matrix $\mathbf{G}$. In [18], $\mathbf{G}$ is chosen to be a Gaussian random matrix. The algorithm is summarized below in Algorithm 1 (we use MATLAB notation). To avoid the need to precompute the matrix rank, we use an error estimate (Algorithm 1). If the complexity of the matvec is $\mu(m,n)$, then the complexity of the algorithm is $\mathcal{O}(\mu(m,n)\ell + \mu(n,m)r + m\ell^2 + nr^2)$. Assuming $m < n$, the complexity is $\mathcal{O}(\mu(m,n)\ell + nr^2)$. If we have a dense matrix, the complexity is $\mathcal{O}(mn\ell)$. It follows that the overall complexity of this approximate factorization is $\mathcal{O}(\ell nm)$ for work and $\mathcal{O}(rm + rn)$ for storage.

4.2 Recursive SVD

Let $\mathbf{M}_1$ and $\mathbf{M}_2$ be two matrices of size $m \times n$. We wish to construct the approximate SVD of

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}_1 \\ \mathbf{M}_2 \end{bmatrix} \tag{9}$$

assuming we know the approximate SVDs of $\mathbf{M}_1$ and $\mathbf{M}_2$. Let $\mathbf{M}_1 = \mathbf{\Phi}_1\mathbf{\Lambda}_1\mathbf{\Psi}_1^*$ and $\mathbf{M}_2 = \mathbf{\Phi}_2\mathbf{\Lambda}_2\mathbf{\Psi}_2^*$. For notational simplicity let r be the rank of the approximations for both $\mathbf{M}_1$ and $\mathbf{M}_2$ so that $\mathbf{\Phi}_i \in \mathbb{C}^{r\times r}$, $i = 1, 2$. We seek to compute $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{\Sigma}$ so that $\mathbf{M}_\varepsilon = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$ and $\|\mathbf{M} - \mathbf{M}_\varepsilon\|$ is smaller than a prescribed tolerance.

Algorithm 1 *Randomized SVD*

Inputs: $\mathbf{M} \in \mathbb{C}^{m \times n}$, ε.
Outputs: approximate rank r, $\mathbf{\Phi}, \mathbf{\Lambda}, \mathbf{\Psi}$ such that $\mathbf{M} \approx \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Psi}^*$.
$r = 1$
$\ell = r + 20$
Create $\mathbf{G}$ (Gaussian random matrix $\mathbf{G} \in \mathbb{C}^{\mathbf{n} \times \ell}$)
$\mathbf{R} = \mathbf{M}\mathbf{G}$
$[\mathbf{U_r}, \mathbf{S_r}, \mathbf{V_r}] = \text{SVD}(\mathbf{R})$
$\mathbf{Q}(:, \mathbf{1} : \mathbf{r}) = \mathbf{U_r}(:, \mathbf{1} : \mathbf{r})$
error estimate $= \|\mathbf{M}\mathbf{G} - \mathbf{Q}\mathbf{Q}^\star\mathbf{M}\mathbf{G}\|$
if error estimate $> \varepsilon \times \mathbf{S_r}(\mathbf{1}, \mathbf{1})$ **then**
    Increase r (e.g., $r = r + 0.05m$)
    Goto 4
end if
$\mathbf{T} = \mathbf{M}^\star\mathbf{Q}$
$[\mathbf{\Phi}, \mathbf{\Lambda}, \mathbf{W}] = \text{SVD}(\mathbf{T}^\star)$
$\mathbf{\Psi} = \mathbf{Q}\mathbf{\Phi}$

First, observe that $\mathbf{M}$ can be written as

$$\mathbf{M} = \mathbf{\Phi}\mathbf{T}, \quad \text{where } \mathbf{\Phi} = \begin{bmatrix} \mathbf{\Phi}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi}_2 \end{bmatrix} \in \mathbb{C}^{2m \times 2r} \text{ and } \mathbf{T} = \begin{bmatrix} \mathbf{\Lambda}_1\mathbf{\Psi}_1^* \\ \mathbf{\Lambda}_2\mathbf{\Psi}_2^* \end{bmatrix} \in \mathbb{C}^{2r \times n}. \tag{10}$$

We can apply the randomized SVD algorithm to $\mathbf{T}$ to compute $\mathbf{T} = \mathbf{U}_T\mathbf{\Sigma}_T\mathbf{V}_T^*$. Then, the SVD factors of $\mathbf{M}$ are $\mathbf{U} = \mathbf{\Phi}\mathbf{U}_T$, $\mathbf{\Sigma} = \mathbf{\Sigma}_T$, and $\mathbf{V} = \mathbf{V}_T$.

Remark. The SVDs of $\mathbf{M}_i$ are not necessary. All we need are $\mathbf{\Phi}_1$ and $\mathbf{\Phi}_2$, so that

$$\mathbf{T} = \begin{bmatrix} \mathbf{\Phi}_1^*\mathbf{M}_1 \\ \mathbf{\Phi}_2^*\mathbf{M}_2 \end{bmatrix}$$

and the algorithm proceeds by computing the approximate SVD of $\mathbf{T}$.

Complexity. The complexity of computing the SVD of $\mathbf{T}$ is $2Rrn$, where $R \leq 2r$ the numerical rank of $\mathbf{T}$. The cost of applying $\mathbf{M}$ on a vector is $\mathcal{O}(mr) + \mathcal{O}(Rn)$. The overall storage requirements are also $\mathcal{O}(mr) + \mathcal{O}(Rn)$.

Accuracy. We will assume that $\mathbf{\Phi}_i$ are known approximately, that is $\|\mathbf{\Phi}_i\mathbf{\Phi}_i^*\mathbf{M}_i - \mathbf{M}_i\| \leq \sigma_{r+1}$ and $\mathbf{M}_i = \mathbf{M}_{\varepsilon,i} + \mathbf{E}_i$. Let $\mathbf{M} = \mathbf{M}_\varepsilon + \mathbf{E}$ indicate the approximation of $\mathbf{M}$ due to approximation errors in the $\mathbf{M}_i$ blocks. Let $\mathbf{U}$ be the approximate range-space basis for $\mathbf{M}_\varepsilon$ computed by the recursive SVD algorithm. Then,

$$\begin{aligned} \|\mathbf{U}\mathbf{U}^*\mathbf{M} - \mathbf{M}\| &= \|\mathbf{U}\mathbf{U}^*(\mathbf{M}_\varepsilon + \mathbf{E}) - (\mathbf{M}_\varepsilon + \mathbf{E})\| \\ &\leq \|\mathbf{U}\mathbf{U}^*\mathbf{M}_\varepsilon - \mathbf{M}_\varepsilon\| + \|\mathbf{U}\mathbf{U}^*\mathbf{E} - \mathbf{E}\| \\ &\leq \sigma(\mathbf{M}_\varepsilon)_{R+1} + 2\sigma_{r+1} \\ &\leq \sigma(\mathbf{M})_{R+1} + 2\sigma_{r+1} \end{aligned}$$

where the last line follows by $\sigma(\mathbf{M}_\varepsilon)_{R+1} \approx \sigma(\mathbf{M})_{R+1}$, due to the stability of singular values due to small perturbations. Indeed, from Theorem 8.6.2 in [15] (also see [30, 31]) the error between the singular values of $\mathbf{A}+\mathbf{E}$ and the ones of $\mathbf{A}$ is bounded, i.e. $|\sigma_k(\mathbf{A}+\mathbf{E}) - \sigma_k(\mathbf{A})| \leq \|\mathbf{E}\|_2, \forall k$. Therefore, assuming that $\sigma(\mathbf{M})_{R+1} \approx \sigma_{r+1}$, we get

$$\|\mathbf{U}\mathbf{U}^*\mathbf{M} - \mathbf{M}\| \leq 3\sigma_{r+1}. \tag{11}$$

Hence if we have L levels of recursion, we observe a factor of $L+1$ larger error. In our experiments, we did not observe this dependency and the recursive SVD is as accurate as applying the randomized SVD directly to $\mathbf{M}$. As mentioned in the introduction, classical dense linear recursive QR factorizations can be used instead [12]. Using those methods the approximation of the range space of $\mathbf{M}$ error does not depend on the recursion depth, but they are harder to implement and parallelize (something that we do not discuss here).

5 FaIMS

Using the randomized and recursive SVD we can now describe the complete algorithm for the reconstruction of $\boldsymbol{\eta}$. We recall that, for a fixed frequency ω, the forward problem is given by

$$\boldsymbol{\phi}^\omega_{ds} = \sum_{j=1}^{N} \mathbf{G}^\omega_{dj} \boldsymbol{\eta}_j \mathbf{u}^\omega_{js}, \quad d = 1, \ldots, N_d, \quad s = 1, \ldots, N_s, \tag{12}$$

where $\boldsymbol{\phi}^\omega_{ds}$ is the set of measurement at the detector locations. We write (12) using a matrix notation as $\boldsymbol{\phi}^\omega_s = \mathbf{M}^\omega_s \boldsymbol{\eta}$ where $\mathbf{M}^\omega_s = \mathbf{G}^\omega \operatorname{diag}(\mathbf{u}^\omega_s)$. The overall algorithm can be stated as follows:

- STEP A: For each frequency, reduce the number of incident fields.
- STEP B: For each frequency, compute an approximate range-space basis for $\mathbf{G}^\omega$ using Algorithm 1.
- STEP C: Combine the approximations of $\mathbf{M}^\omega_s$ to build an approximation for $\mathbf{M}^\omega$.
- STEP D: Combine $\mathbf{M}^\omega$ to build an overall approximation of $\mathbf{M}$ using the recursive SVD.

Inputs. We specify the domain (unit cube) size L defined in wavelengths λ units (H is the domain $[0; L]^3$), the source locations, the detector locations, the number of points N that will be used to discretize H, and the incident wave frequencies $\omega_{1,\ldots,N_\omega}$.

STEP A: **Reducing the number of incident fields.** This is a preprocessing stage that requires the evaluation of $\mathbf{u}^\omega(\mathbf{r})$ at the scatterer positions. The reduction is done separately for each frequency ω. Let us introduce the approximate SVD of $\mathbf{u}^\omega = \boldsymbol{\Phi}^\omega \boldsymbol{\Lambda}^\omega \boldsymbol{\Psi}^{\omega *}$ (where $\mathbf{u}^\omega = [\mathbf{u}^\omega_1 \ldots \mathbf{u}^\omega_{N_s}]^t \in \mathbb{C}^{N_s \times N}$) with rank

$R_s^\omega \leq N_s$. Then (12) becomes

$$\begin{aligned}
\boldsymbol{\phi}_{ds}^\omega &= \sum_{j=1}^{N} \mathbf{G}_{dj}^\omega \boldsymbol{\eta}_j \mathbf{u}_{js}^\omega, \quad d = 1, \ldots, N_d, \quad s = 1 \ldots N_s \\
&= \sum_{j=1}^{N} \mathbf{G}_{dj}^\omega \boldsymbol{\eta}_j \sum_{l'=1}^{R_s^\omega} \boldsymbol{\Lambda}_{l'}^\omega \boldsymbol{\Phi}_{l'j}^\omega \boldsymbol{\Psi}_{l's}^\omega \\
\sum_{s=1}^{N_s} \boldsymbol{\Psi}_{ls}^{\omega *} \boldsymbol{\phi}_{ds}^\omega &= \sum_{j=1}^{N} \mathbf{G}_{dj}^\omega \boldsymbol{\eta}_j \sum_{l'=1}^{R_s^\omega} \boldsymbol{\Lambda}_{l'}^\omega \boldsymbol{\Phi}_{l'j}^\omega \sum_{s=1}^{N_s} \boldsymbol{\Psi}_{ls}^{\omega *} \boldsymbol{\Psi}_{l's}^\omega \\
\hat{\boldsymbol{\phi}}_{dl}^\omega &= \sum_{j=1}^{N} \mathbf{G}_{dj}^\omega \boldsymbol{\eta}_j \boldsymbol{\Lambda}_l^\omega \boldsymbol{\Phi}_{lj}^\omega, \\
\hat{\boldsymbol{\phi}}_{dl}^\omega &= \sum_{j=1}^{N} \mathbf{G}_{dj}^\omega \boldsymbol{\eta}_j \hat{\mathbf{u}}_{jl}^\omega, \quad d = 1, \ldots, N_d, \quad l = 1 \ldots R_s^\omega.
\end{aligned}$$

In other words, we use the orthonormality of the $\boldsymbol{\Psi}^\omega$ and transform the incident fields *and* the data to obtain

$$\boxed{\hat{\boldsymbol{\phi}}_{dl}^\omega \approx \mathbf{G}^\omega \text{ diag } (\hat{\mathbf{u}}_{jl}^\omega) \boldsymbol{\eta}}, \; 1 \leq l \leq R_s^\omega, \quad 1 \leq \omega \leq N_\omega, \tag{13}$$

where $\hat{\boldsymbol{\phi}}_{dl}^\omega = \boldsymbol{\Psi}_l^{\omega *} \boldsymbol{\phi}^\omega$ and $\hat{\mathbf{u}}_l^\omega = \boldsymbol{\Lambda}_l^\omega \boldsymbol{\Phi}_l^\omega$. Here $\boldsymbol{\Phi}_l^\omega$ denotes the l-th column of the matrix $\boldsymbol{\Phi}^\omega \in \mathbb{C}^{N \times R_s^\omega}$ and $\boldsymbol{\Psi}_l^\omega$ denotes the l-th column of the matrix $\boldsymbol{\Psi}^\omega \in \mathbb{C}^{N_s \times R_s^\omega}$. As we increase the number of sources, for fixed noise we expect that the rank R_s^ω will be much smaller than N_s. [6]

STEP B: **Computation of the randomized SVD of $\mathbf{G}^\omega$.** For each frequency ω, we compute the approximate SVD of the matrix $\mathbf{G}^\omega$, $\mathbf{G}_\varepsilon^\omega = \boldsymbol{\Phi}_g^\omega \boldsymbol{\Lambda}_g^\omega \boldsymbol{\Psi}_g^{\omega *} \in \mathbb{C}^{N_d \times N}$ using the randomized SVD (section 4.1). We use R_g^ω to denote the rank of $\mathbf{G}_\varepsilon^\omega$; R_g^ω depends on the approximation tolerance for the SVD and satisfies $R_g^\omega \leq N_d, N$.

STEP C: **Combine the SVDs of all sources for a fixed frequency.** To combine the approximate SVDs of the R_s sources for a fixed frequency, we compute the approximate SVD of

$$\begin{bmatrix} \mathbf{G}^\omega \mathbf{D}_1^\omega \\ \vdots \\ \mathbf{G}^\omega \mathbf{D}_{R_s}^\omega \end{bmatrix} \in \mathbb{C}^{R_s N_d \times N}, \tag{14}$$

where $\mathbf{G}^\omega \in \mathbb{C}^{N_d \times N}$ and $\mathbf{D}_\ell^\omega = \text{diag}(\hat{\mathbf{u}}_\ell^\omega) \in \mathbb{C}^{N \times N}$. Instead of computing directly the approximate SVD of this large matrix, we use the approximate SVD of $\mathbf{G}^\omega$ computed during step B; $\mathbf{G}_\varepsilon^\omega = \boldsymbol{\Phi}_g^\omega \boldsymbol{\Lambda}_g^\omega \boldsymbol{\Psi}_g^{\omega *}$. More precisely, we compute the approximate SVD of

$$\mathbf{B}^\omega = \begin{bmatrix} \boldsymbol{\Lambda}_g^\omega \boldsymbol{\Psi}_g^{\omega *} \mathbf{D}_1^\omega \\ \vdots \\ \boldsymbol{\Lambda}_g^\omega \boldsymbol{\Psi}_g^{\omega *} \mathbf{D}_{R_s}^\omega \end{bmatrix} \in \mathbb{C}^{R_s R_g^\omega \times N}$$

[6] A similar approach could be followed for $\boldsymbol{\phi}_{ds}$ to guide numerical rank selection in the building of the approximate SVD of $\mathbf{M}$.

We compute $\mathbf{B}_\varepsilon^\omega$, the approximate SVD of $\mathbf{B}^\omega$ with the randomized SVD. R_ω denotes its approximate rank. Finally, we transform the data accordingly, i.e. we replace

$$\begin{bmatrix} \mathbf{d}_\mathbf{1}^\omega \\ \vdots \\ \mathbf{d}_{\mathbf{R_s}}^\omega \end{bmatrix} \in \mathbb{C}^{R_s N_d} \text{ by } \begin{bmatrix} \mathbf{\Phi}_g^{\omega *} \mathbf{d}_\mathbf{1}^\omega \\ \vdots \\ \mathbf{\Phi}_g^{\omega *} \mathbf{d}_{\mathbf{R_s}}^\omega \end{bmatrix} \in \mathbb{C}^{R_s R_g^\omega}. \tag{15}$$

This transformation is beneficial when applying $\mathbf{G}^\omega$ to a vector has a work complexity that is larger than applying the approximate factorization of $\mathbf{G}^\omega$.

STEP D: **Recursion over frequencies.** In the last step, we combine the N_ω approximate SVDs of $\mathbf{M}^\omega$ (corresponding to N_ω frequencies) computed in step C. We apply the recursive SVD (section 4.2). Let R denote the number of selected singular values (smaller than a prescribed tolerance). We define $\mathbf{T}$, $\mathbf{W}$ and $\mathbf{S}$ such that $\mathbf{M}_\varepsilon = \mathbf{TSW}^\star$. $\mathbf{T}$ and $\mathbf{W}$ are two matrices of size $N_\omega R_s R_g^\omega \times R$ and $N \times R$ respectively. $\mathbf{S}$ is a diagonal matrix of size R. Instead of applying directly the recursive SVD to combine N_ω frequencies, we apply recursively this algorithm to combine two frequencies at each level of the recursion tree (see Fig. 3). One possible optimization would be to omit visiting all the leaves of the frequency tree (guided by changes in the approximate rank) to reduce the number of frequencies for which we need to solve the problem.

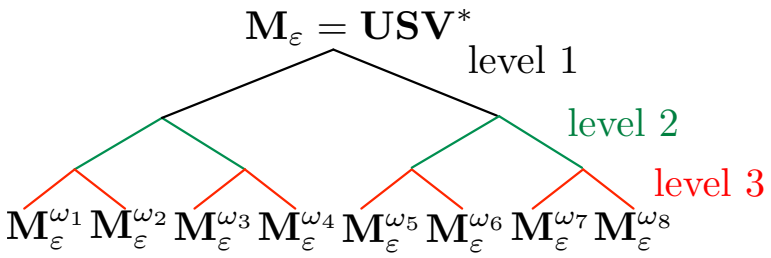


Figure 3: *Instead of applying directly the recursive SVD (section 4.2) on N_ω frequencies, we apply the algorithm recursively to combine two frequencies at each level of the recursion tree (this example is given for the case $N_\omega = 8$).*

This completes the calculation of the approximation of $\mathbf{M}$. Note that the transformation of the data is not necessary and desired if we want to apply the approximate $\mathbf{M}$ to a vector. The modification of the algorithm so that the matvec is applied quickly without transforming the data is straightforward. Finally, the construction of the pseudoinverse is immediate.

Overall complexity estimate for work and storage. Let $\mu(N)$ be the cost of solving a forward or adjoint Helmholtz problem assuming $N_s \leq N$ and $N_d \leq N$. Also let $R_s = \max_\omega R_s^\omega$.

- STEP A: The cost of reducing the number of incident fields is the cost for the construction of the low rank SVDs for matrices of size $N_s \times N$ for each frequency ω. The cost of a single frequency is $\mathcal{O}(R_s \mu(N))$ and the overall cost is $\mathcal{O}(R_s N_\omega \mu(N))$ (where $R_s \leq N_s$).
- STEP B: The cost of the computation of the randomized SVD of $\mathbf{G}^\omega \in \mathbb{C}^{N_d \times N}$ for a fixed frequency is $\mathcal{O}(R_g^\omega \mu(N))$ so that the total cost of this step is $\mathcal{O}(R_g^\omega N_\omega \mu(N))$ (where $R_g^\omega \leq N_d$).

- STEP C: The cost of combining the sources is reduced to the cost of the computation of the randomized SVD of $\mathbf{B}^\omega \in \mathbb{C}^{R_s R_g^\omega \times N}$: $\mathcal{O}(R_\omega R_s R_g^\omega N)$ for each frequency (with $R_\omega \leq \min(R_s R_g^\omega, N)$). So that the total cost of this step is $\mathcal{O}(R_\omega R_s N_\omega R_g^\omega N)$.

- STEP D: Let R be the approximate rank of $\mathbf{M}$. Let $R_g = \max_\omega R_g^\omega$. Let $L = \log N_\omega$ and let R_ℓ be the rank of the operator at a node at level ℓ (assuming for simplicity that all such nodes have the same rank) and $R_L = R_\omega$. For the leaves of frequency tree, the cost per leaf is $R_L R_s \mu(N)$ or $R_L R_s R_g N$. The total cost for the leaves is $N_\omega R_L R_s R_g N$. For an internal node, the cost to combine the SVDs of its children is $R_\ell(2R_{\ell+1}N)$ and the total cost at level ℓ is $2^\ell R_\ell(2R_{\ell+1}N)$. The worst case is there is not compression as we traverse the tree or equivalently $R_\ell \approx (R/2^\ell)$. Then the overall cost of traversing the tree (omitting constants) is $\sum_{\ell=1}^{L-1} 2^\ell R_\ell(2R_{\ell+1}N) \leq \mathcal{O}(R^2 N \log N_\omega)$.

Therefore the overall complexity of FaIMS is

$$\boxed{\mathcal{O}((R_s + R_g)N_\omega \mu(N)) + \mathcal{O}(R_\omega R_s N_\omega R_g N) + \mathcal{O}(R^2 N \log N_\omega).}$$

Note that applying the randomized SVD algorithm directly to $\mathbf{M}$ has cost $\mathcal{O}(RN_s N_\omega \mu(N)) + \mathcal{O}(R^2 N)$. Since $R_\omega \leq R$, $R_s \leq N_s$ and assuming $R_g N \leq \mu(N)$, the overall complexity of FaIMS same as applying a randomized SVD directly to $\mathbf{M}$ up to the $\log N_\omega$ factor. (If $R \log N_\omega \leq N_s N_\omega \mu(N)/N$, then the complexity is similar.)

However, FaIMS provides better locality in the calculations (and thus, better potential for parallel scalability). Most important if we do not have an optimal solver (i.e., $\mu(N) \gg R_g N$) FaIMS will be faster than direct SVD since it requires $N_\omega(R_s + R_g)$ solutions of a scattering problem as opposed to $N_s N_\omega R$.

In each of the four steps of FaIMS, we perform a low rank approximation of the system matrix. Because this matrix is defined by the values of a function on a given discretization, its rank is constant even though we use a coarser discretization. As a result for large enough numbers of sources, detectors, excitation frequencies and/or discretization of H, the values of R_s, R_g, R_ω and R are constant. As a result the final complexity estimate becomes

$$\mathcal{O}(\mu(N)N_\omega + N_\omega \mu(N)).$$

As we remarked before, one can also reduce the N_ω factor by avoiding vising all the leaves of the tree. One can use a greedy method in which new frequencies are added only if there is sufficient change in the rank. We have not explored this approach in this paper.

Storage. For the step A, the storage of the N_ω singular vectors is order $N_\omega R_s N$. For the step B, the storage of the N_ω singular vectors is order $N_\omega R_g N$. For the step C, the cost is $N_\omega R_\omega N$. Finally, the storage cost of the combined matrices in step D is $\mathcal{O}(RN)$. The storage of the singular vector of the approximate SVD of $\mathbf{M}$ is also $\mathcal{O}(RN)$.

6 Numerical experiments

We present several source-receiver-scatterer configurations to illustrate the performance of FaIMS. We demonstrate the capability of placing detectors and sources on arbitrary geometries and the overall scalability of the proposed algorithm. We verify the accuracy of the algorithm using two scatterer geometries: a cross-like planar geometry of point scatterers and a biplane-like 3D geometry of point scatterers. We generate the scattered field using single-scattering forward problem approximations (Born Approximation). In each example, the wave velocity of the background medium is set to one. We are focused on demonstrating

the scalability aspects of our algorithm, so we do not consider noise or regularization. The algorithm has been implemented in MATLAB and our experiments took place on a AMD Opteron workstation. The tolerance ε in the randomized SVD algorithm has been set to 10^{-9} in all of our experiments. The truncation parameter (regularization) we used to invert the approximate factorization of $\mathbf{M}$, was $10^{-8}\sigma_1$, where σ_1 is the maximum singular value of $\mathbf{M}$.

6.1 Description of the test problems

Definition of the main parameters used in our tests.

- The N_ω *incoming field frequencies* are equispaced in the $[\omega_{\min}, \omega_{\max}]$ interval.
- *Length scales* are measured in terms of the smallest wavelength $\lambda := 2\pi/\omega_{\max}$. (The higher the frequency, the lower the compression of the operators and thus, the higher the computational cost of the inversion.) We also use the parameter $L = 12\lambda$.

Specification of the target medium perturbations and parameters for the different numerical experiments.

- **Cross-like geometry.** We consider two test problems based on a simple cross-like geometry located at the $z = 6\lambda$ plane.
  - For the Cross A (Fig. 4A), the detectors are regularly spaced on the plane $z = L$ and the sources on the plane $z = 0$.
  - For the Cross B, the detectors and sources are located on arbitrary geometries (Fig. 4B).

  We verify the accuracy of our approximate SVD reconstruction on the Cross A scatterer model with size λ. We use four sources with multiple excitation frequencies to generate the incident fields and measure the scattered fields at $[21]^2$ detectors. The computational domain is discretized with a linear grid with size $[21]^2$. Also on these two test problems, we verify the scalability of the algorithm with increasing N, N_d, N_s and N_ω[7] and demonstrate the effectiveness of a low rank approximation. See Table 3 for a summary of the problem and scatterer model sizes used in this set of experiments.

N	$\omega_{\min}$	$\omega_{\max}$	N_ω	N_s	N_d	scatterer size (λ)
11^2	50	100	8	16	10^2	0.01 and 1
21^2	50	100	16	64	20^2	0.01 and 1
41^2	50	100	32	256	40^2	0.01 and 1
81^2	50	100	64	1024	80^2	0.01 and 1

Table 3: *Cross-like geometry: Summary of the parameters used to verify the scalability of FaIMS. In the following, we will refer to one of those four tests by the mesh size, i.e., $N = 11^2$, $N = 21^2$, $N = 41^2$ or $N = 81^2$.*

- **Biplane geometry.** We consider a test problem with a more complex geometry. The sources and detectors are located on a sphere (Fig. 4D). We consider two scatterer model sizes: λ and 5λ. We use 162 detectors, 12 sources, 8 frequencies and we set $N = 51^3$.

[7] Given a minimum and a maximum frequency, we verify that the algorithm scales well as we increase the number of sample frequencies in this range.

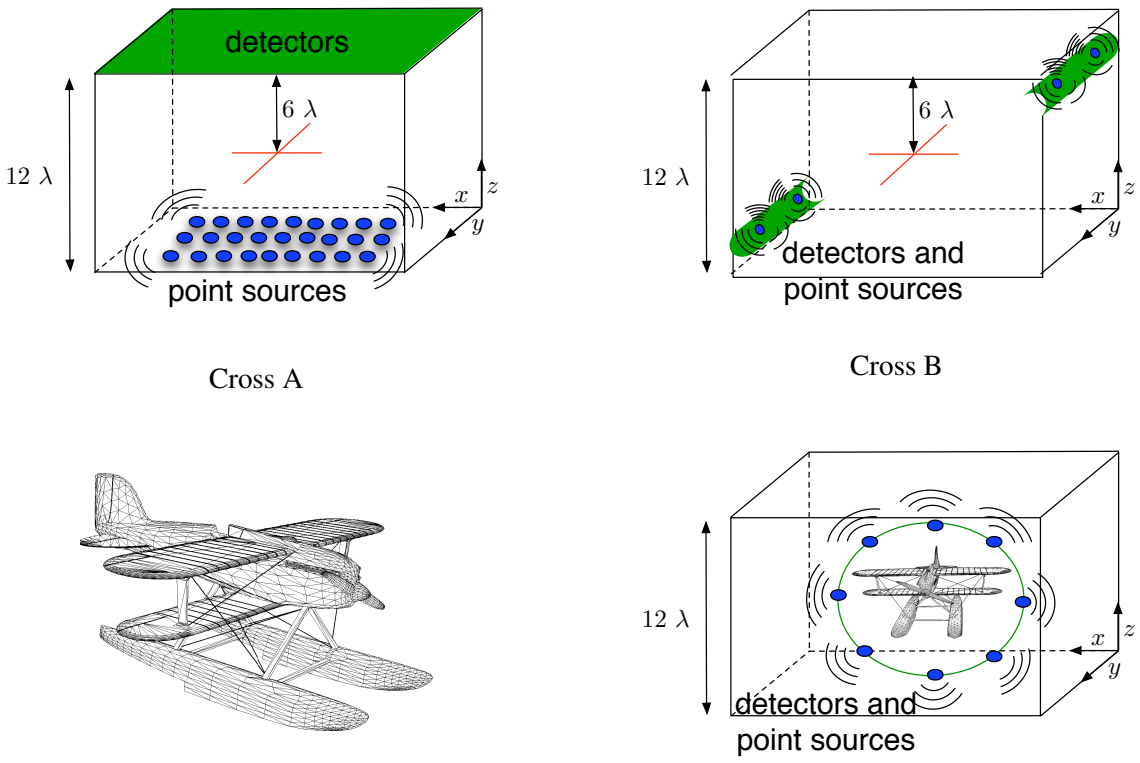


Figure 4: *Definition of the three test problems we have used to test FaIMS. We use two geometries of point scatterers : a cross (*A *and* B*) and a biplane. For the cross* A*, we generate the incident field by sources and detectors regularly spaced on the planes $z = 0$ and $z = 12\lambda$ respectively. For the Cross* B *test problem, the detectors and point sources are located on two arbitrary geometries. For the biplane-like geometry, the sources and detectors are located on a sphere. The triangulation of the biplane geometry is used for visualization only. To generate the data, we compute the scattered field due to point scatterers located at the vertices of the mesh.*

6.2 Results

Cross A. We first verify the accuracy of our approximate SVD reconstruction. In Table 4, we report the relative error on the approximation normalized by the value of the maximum singular value. We also compute the relative error between FaIMS and the `SVD` MATLAB function on the singular values (normalized by the value of the maximum singular value). Finally, we report the relative error between FaIMS and the `SVD` MATLAB function on the approximation of the inverse.

To verify the efficiency and accuracy of our algorithm, we solve the inverse problem both using our FaIMS algorithm and the `LSQR` MATLAB function. The termination tolerance in the `LSQR` algorithm is set to 10^{-5}. On Fig. 5, we report $\boldsymbol{\eta}$, the results of the inversion, for various mesh sizes and for the scatterer model of the cross geometry with size λ. The left column plots represent the results of FaIMS and the right column plots the results with the `LSQR`. The mesh sizes handled using the `LSQR` function are limited due to

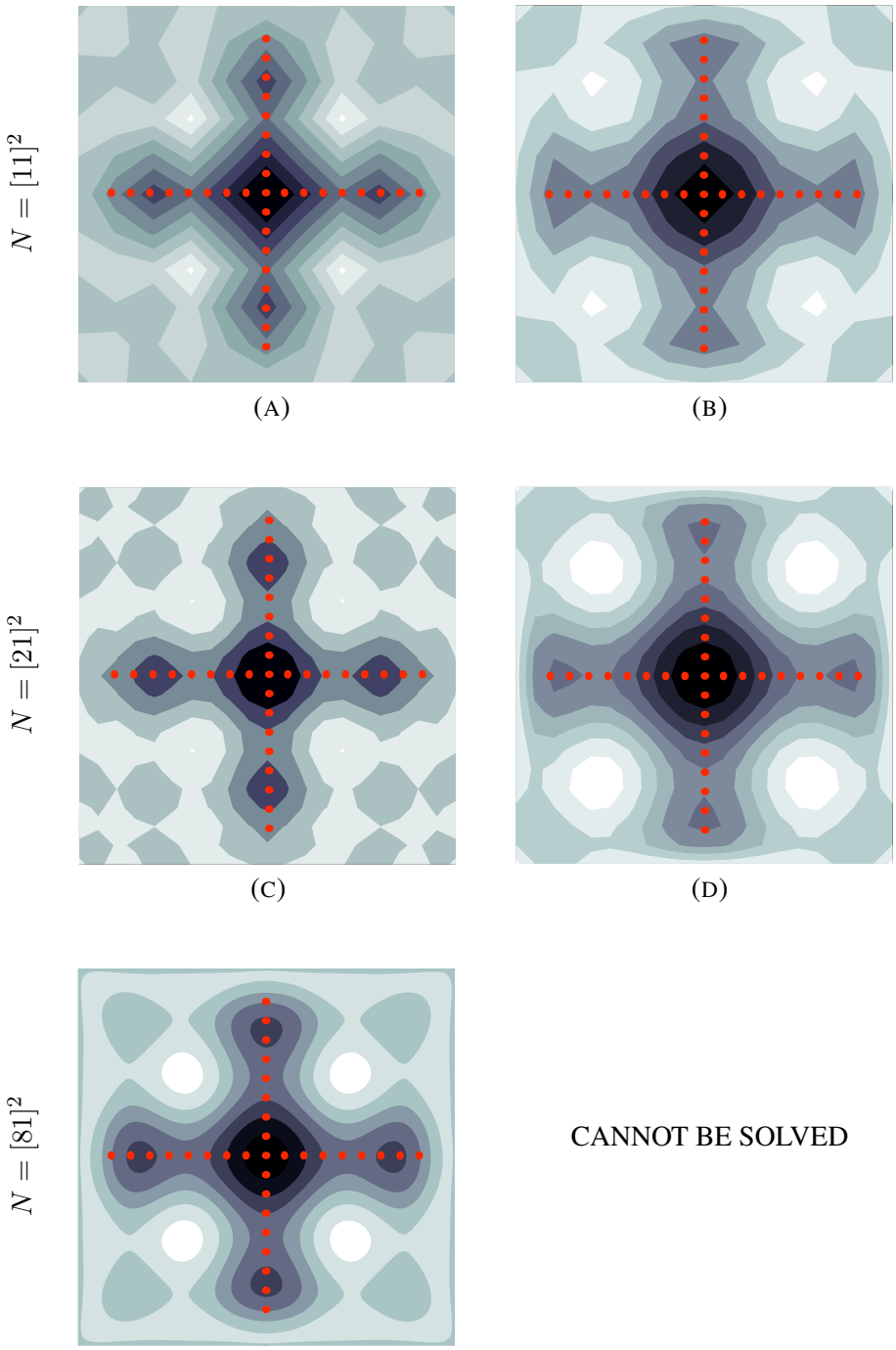


Figure 5: *Cross* A *: We report* $\boldsymbol{\eta}$*, the result of the inversion, at the* $z = 6\ \lambda$ *plane for the scatterer model cross* A *with size* λ*. The left column represents the results using FaIMS (*A,C,E*) and the right column the results using the* LSQR *MATLAB function with tolerance set to* 10^{-5} (B,D)*. The red points represent the true point scatterer locations. The two methods produce very similar results.*

	$N_\omega = 4$	$N_\omega = 8$	$N_\omega = 16$	$N_\omega = 32$
$\|\mathbf{M} - \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^*\|/\sigma_{max}$	$3.07\ 10^{-6}$	$2.34\ 10^{-6}$	$2.10\ 10^{-6}$	$2.01\ 10^{-6}$
$\|\boldsymbol{\Sigma} - \boldsymbol{\Sigma}_{matlab}\|/\sigma_{max}$	$1.00\ 10^{-7}$	$3.12\ 10^{-7}$	$1.73\ 10^{-7}$	$1.90\ 10^{-7}$
$\|\boldsymbol{\eta} - \boldsymbol{\eta}_{matlab}\|/\boldsymbol{\eta}_{matlab}$	$1.72\ 10^{-2}$	$2.29\ 10^{-2}$	$1.19\ 10^{-2}$	$2.95\ 10^{-2}$

Table 4: *We report the relative error on the approximation normalized by the value of the maximum singular value. We also compute the relative error between FaIMS and the* SVD *MATLAB function on the singular values (normalized by the value of the maximum singular value). Finally, we report the relative error between FaIMS and the* SVD *MATLAB function on the approximation of the inverse. The scatterer model is the Cross* A *with size* λ*. We use four sources with multiple excitation frequencies to generate the incident fields and measure the scattered fields at* $[21]^2$ *detectors. The computational domain is discretized using a Cartesian grid of size* $[21]^2$*.*

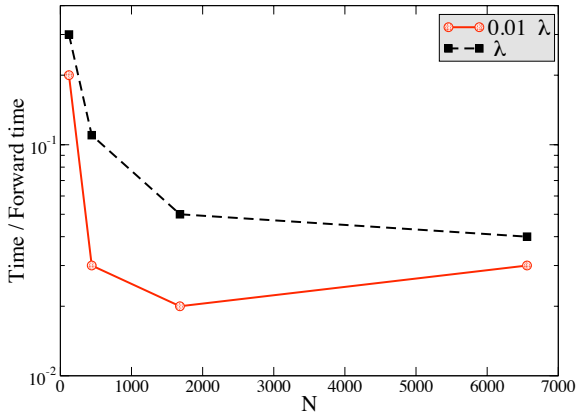
memory constraints.[8] Both methods lead to a good accuracy of the reconstruction. In both cases, the larger the problem is, the better the reconstruction. The accuracy of FaIMS is slightly better. The reason is that we terminated LSQR early.

In Figure 6, we report the CPU time for the four main steps of FaIMS: the reduction of the number of incident fields, the computation of the approximate SVDs of the Green's functions, the combining of the approximate SVDs for all sources for a fixed frequency and the combining of the approximate SVDs for all frequencies, for the two scatterer model sizes (0.01λ and λ). We normalize the CPU time by the total CPU time required to generate the incident field (which is linear in $N_d,\ N_\omega, N_s$ and N). We note that the smaller the cross size is, the smaller the CPU time. This is due to the effectiveness of the low-rank approximation. For the scatterer model with size λ, the major CPU cost is the combining of the approximate SVDs for all sources whereas, as expected, this step takes a small portion of time for the scatterer model with size 0.01λ. Again, as we increase N the normalized CPU time of each step is reduced. This result is in agreement with our complexity estimate.

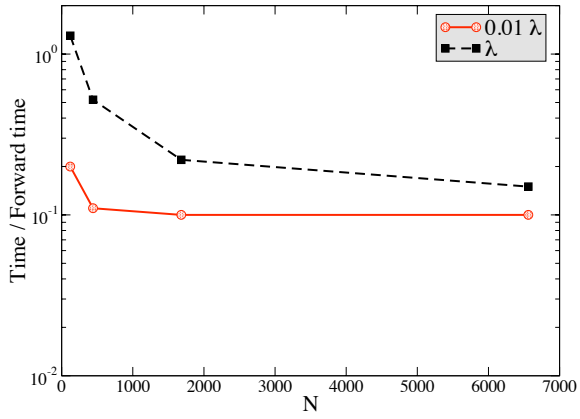
In Table 5, we report the normalized total CPU time of the inversion both with FaIMS and with the LSQR MATLAB function. Again the scatterer model is the Cross A with size 0.01λ and λ. For the LSQR solver, the total inversion time is independent of the scatterer model size whereas FaIMS benefits from low rank approximations at the low frequency regime. FaIMS is clearly faster than the LSQR MATLAB function. In Figure 7, we illustrate the level of compression according to the scatterer model and problem sizes. For each major step of the algorithm, we report the compression rates against the frequency or the level in the tree for the combining of the approximate SVDs for all frequencies. The ranks are normalized by the full rank. Because the inverse problem is ill-posed, the singular values decay very fast. We use a truncated SVD to regularize the formulation.

Cross B. For this second example, the sources and detectors are located on arbitrary geometries. We report the results of the inversion $\boldsymbol{\eta}$ for the case of $N = 81^2$ on Fig. 8 (left: scatterer model with size 0.01λ; right: scatterer model with size λ). This example illustrates the ill-posedness of the problem. For the lowest frequency, only a small number of singular values are selected and the algorithm is very fast. In Figure 9, we report the CPU time for the four main steps of FaIMS: the reduction of the number of incident fields, the computation of the approximate SVDs of the Green's functions, the combining of the approximate SVDs

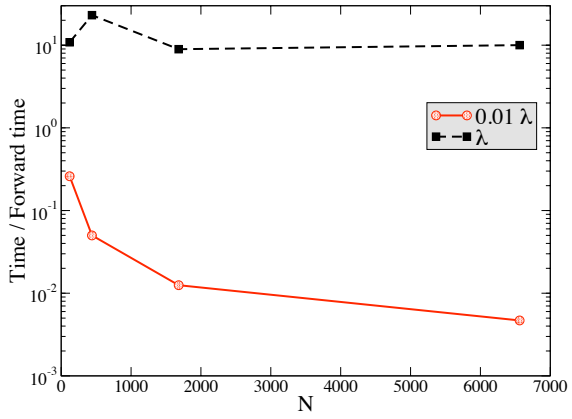
[8] We could also use a simple matrix free approximation. In that case, the limitation is the maximum number of iterations allowed.



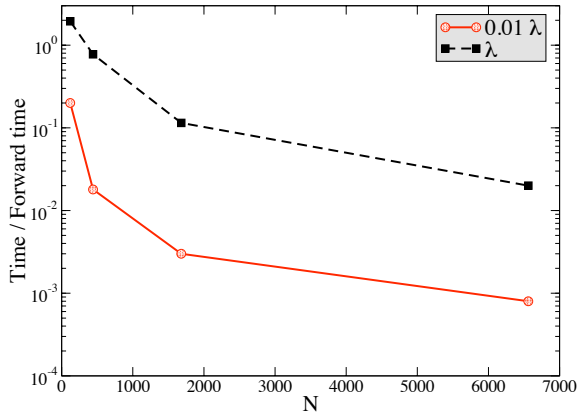
A: Reducing the number of sources



B: approximate SVDs of the Green's functions



C: Combining the approximate SVDs for all sources

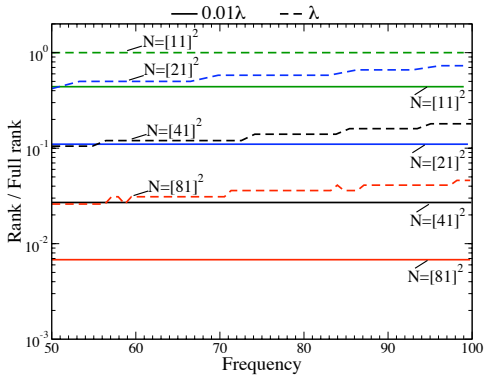


D: Combining the approximate SVDs for all frequencies

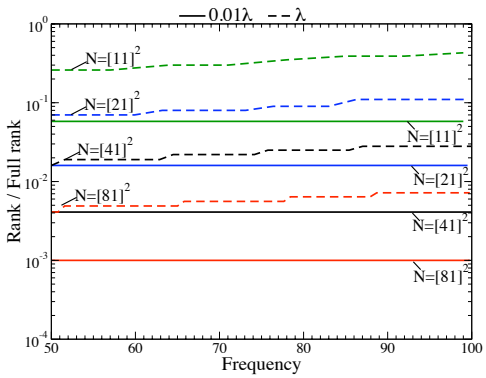
Figure 6: *We report the normalized (by the time to "solve" (evaluate in our case) the forward problem) CPU time for each of the main steps of FaIMS against the mesh size. The scatterer model is the Cross* A *with size* 0.01λ *(plain lines) and* λ *(dashed lines). The smaller is the cross size, the smaller is the normalized CPU time for each step. This is due to the low rank approximation at the low frequency regime. For the scatterer model with size* λ*, the major CPU consuming step is the combining of the SVDs for all sources whereas, as expected, this step consumes only a small portion of time for the scatterer model with size* 0.01λ*.*

		$N = [11]^2$	$N = [21]^2$	$N = [41]^2$	$N = [81]^2$
0.01λ	FaIMS	1.03	0.23	0.14	0.13
	LSQR	11.6 (1)	413.6 (1)	X	X
λ	FaIMS	15.0	24.7	9.3	10.2
	LSQR	13.1 (14)	407.7 (11)	X	X

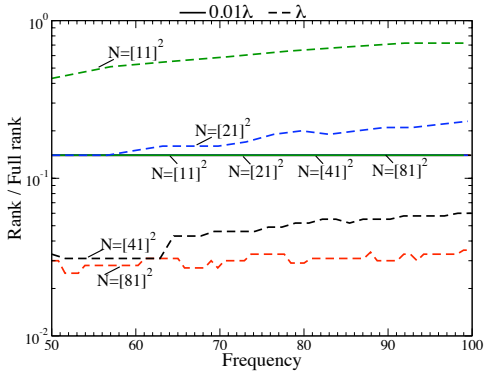
Table 5: *We report the normalized (by the time to solve the forward problem) CPU time against the mesh size using FaIMS and using the* LSQR *MATLAB function. For the* LSQR *we also report the number of iterations (number in parentheses).*



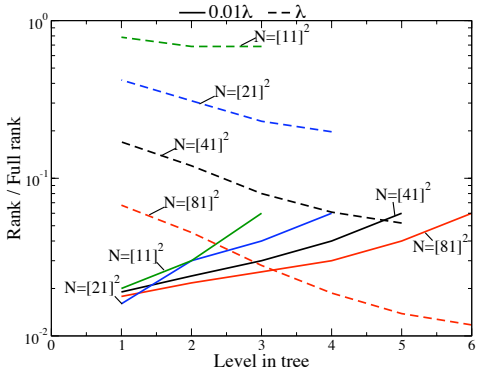
A: Reducing the number of sources



B: approximate SVDs of the Green's functions



C: Combining the approximate SVDs for all sources



D: Combining the approximate SVDs for all frequencies

Figure 7: *Cross* A *: For each major step of the algorithm, we report the compression rates against the frequency or the level in the tree. The ranks are normalized by the full rank. The scatterer model is the Cross* A *with size* 0.01λ *(plain lines) and* λ *(dashed lines). We report the results for the four mesh sizes (red:* $N = 81^2$*, black:* $N = 41^2$*, blue:* $N = 21^2$ *and green:* $N = 11^2$*). Because the useful information is limited, the rank is not dependent of the mesh size. As a result, the larger the mesh is, the larger the compression. Moreover, the larger the scatterer model is, the smaller the compression. Finally, we remark that at each step, the algorithm keeps compressing the information.*

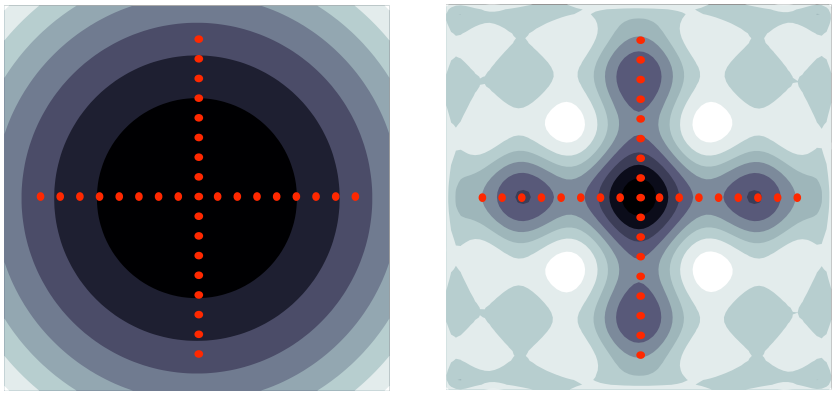


Figure 8: *We report $\boldsymbol{\eta}$ at the $z = 6\ \lambda$ plane. The scatterer model is the Cross* B *with size 0.01λ (left) and λ (right). The* 6561 *detectors are located on the geometry presented on Fig. 4*B*. We generate the data using incident fields generated by* 1024 *sources on the geometry presented on Fig. 4*B *and* 64 *frequencies.*

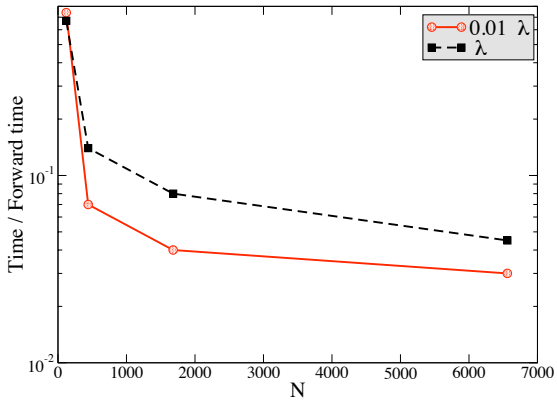
		$N = [11]^2$	$N = [21]^2$	$N = [41]^2$	$N = [81]^2$
0.01λ	FaIMS	4.14	0.35	0.21	0.25
	LSQR	6.4 (1)	295.6 (2)	X	X
λ	FaIMS	18.2	44.5	12.4	14.5
	LSQR	9.5 (49)	309.8 (71)	X	X

Table 6: *We report the normalized (by the time to solve the forward problem) CPU time against the mesh size using FaIMS and using the* LSQR *MATLAB function. For the* LSQR *MATLAB function we also report the number of iterations (number in parentheses).*

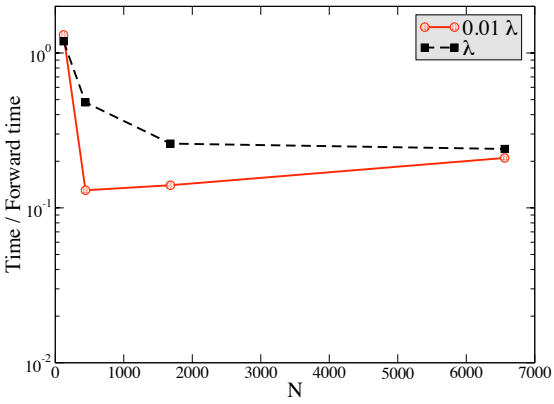
for all sources for a fixed frequency and the combining of the approximate SVDs for all frequencies, for the two scatterer model sizes (0.01λ and λ). We normalize the CPU time by the total CPU time to generate the incident field (which is linear in N_d, N_ω, N_s and N). The smaller the cross size is, the smaller the CPU time since low-rank approximations are effective. For the scatterer model with size λ, the major CPU consuming step is the combining of the approximate SVDs for all sources whereas, as expected, this step consumes only a small portion of time for the scatterer model with size 0.01λ. In Table 6, we report the normalized total CPU time of the inversion both with FaIMS and with the LSQR function. Again the scatterer model is the Cross B with size 0.01λ and λ. For the LSQR solver, the total inversion time is independent of the scatterer model size whereas FaIMS benefits from low rank approximations at the low frequency regime. FaIMS is clearly faster than the LSQR MATLAB function and than the forward solver at low frequencies.

In Figure 10, we report the level of compression according to the scatterer model and problem sizes.

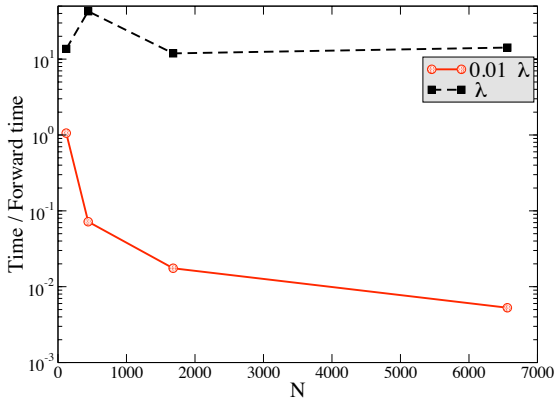
Biplane. This last example demonstrates the ability of the algorithm to recover complex geometries. The scatterer model is a biplane-like geometry with size λ or 5λ. On Figures 11 (size λ) and 12 (size 5λ), we report the results of the inversion $\boldsymbol{\eta}$ under a tabular form. Each row corresponds to a particular plane Oxy,



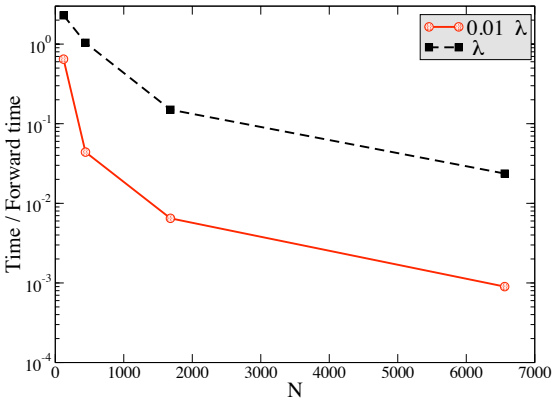
A: Reducing the number of sources



B: approximate SVDs of the Green's functions

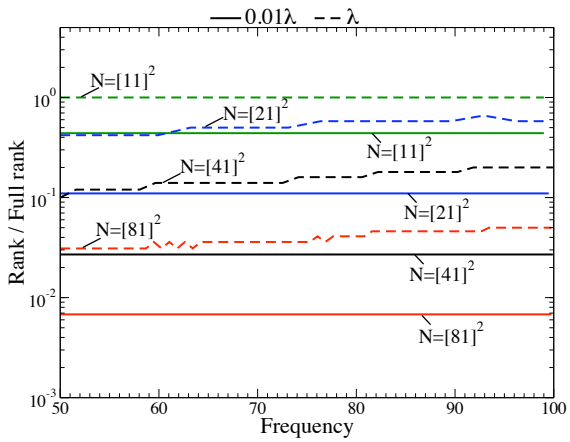


C: Combining the approximate SVDs for all sources

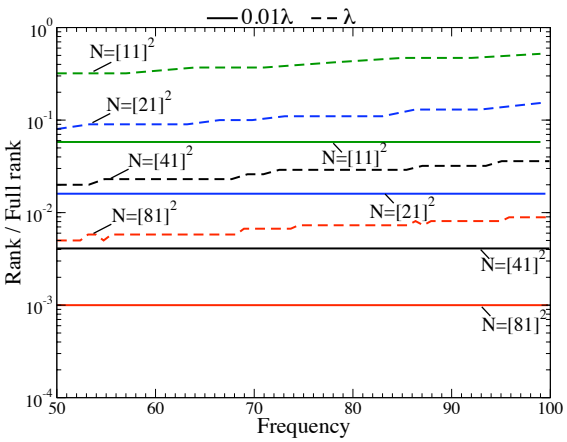


D: Combining the approximate SVDs for all frequencies

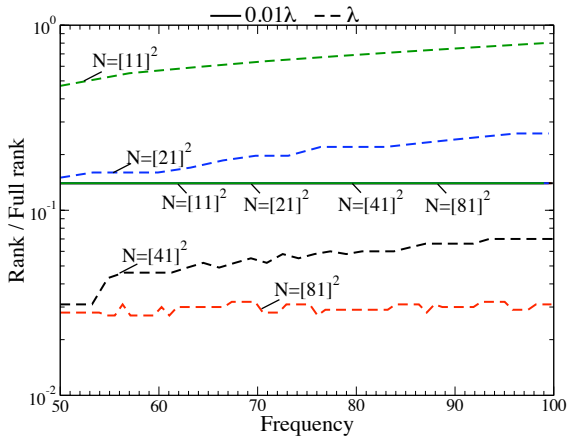
Figure 9: *We report the normalized (by the time to solve the given forward problem) CPU time for each main step of our inversion algorithm against the mesh size. The scatterer model is the Cross* B *with size* 0.01λ *(plain lines) and* λ *(dashed lines).*



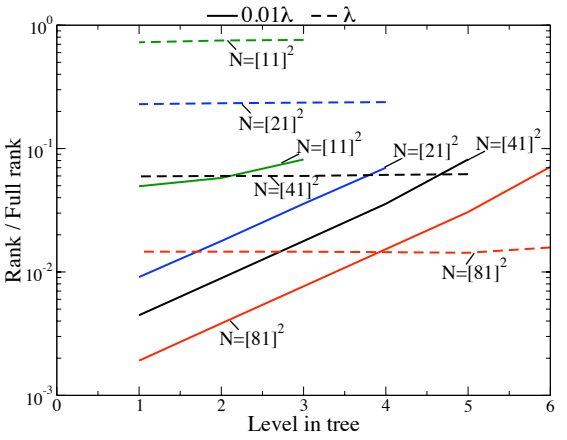
Step A: Reducing the number of sources



Step B: approximate SVDs of the Green's functions



Step C: Combining the approximate SVDs for all sources



D: Combining the approximate SVDs for all frequencies

Figure 10: *Cross B : For each major step of the algorithm, we report the compression rates against the frequency. The ranks are normalized by the full rank. The scatterer model is the Cross B with size* 0.01λ *(plain lines) and* λ *(dashed lines). We report the results for the four mesh sizes (red:* $N = 81^2$*, black:* $N = 41^2$*, blue:* $N = 21^2$ *and green:* $N = 11^2$*).*

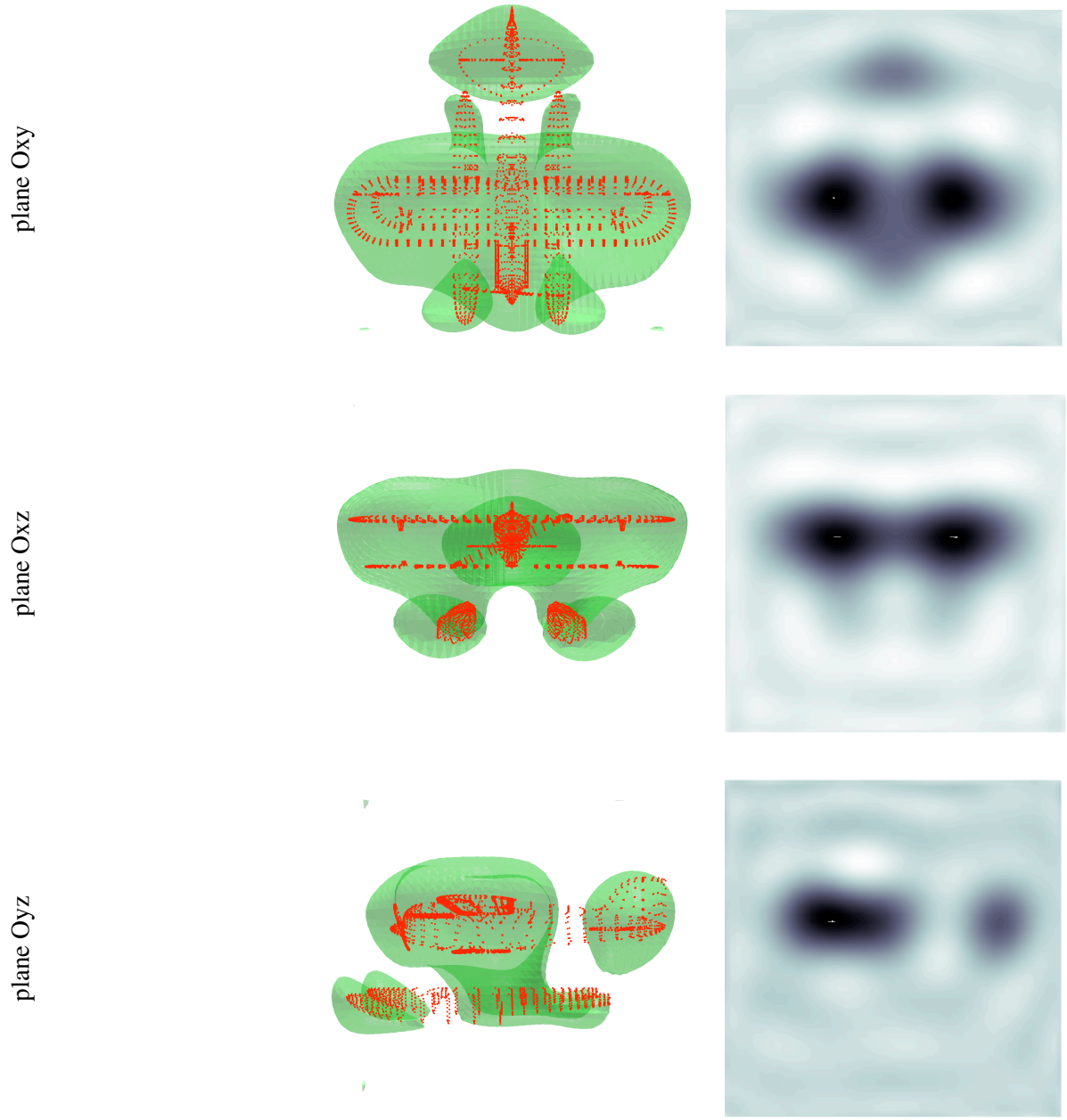


Figure 11: *Results of the inversion for the scatterer model of the biplane with size λ. Each row corresponds to a particular plane Oxy, Oxz or Oyz. The left column corresponds to a 3-D view of the isosurfaces ($\eta = 0.25\eta_{max}$) and the right column corresponds to a 2-D slice, respectively from top to bottom at the planes $z = 0$, $y = 0$ and $x = 0$.*

Oxz or Oyz. The left column corresponds to a 3-D view of the isosurfaces ($\eta = 0.25 * \eta_{max}$) and the right column corresponds to a 2-D slice at the median plane. For the scatterer model with size λ, the inversion is as accurate. We can already distinguish the main components of the biplane. For example, on the Oxz view, we can separate the two floats of the plane. However, we cannot see the two wings or say that the shape is a biplane. Again, this is due to the ill-posedness of the problem at low frequencies. On the other hand, for the scatterer model with size 5λ, the system matrix is nearly full rank. As a result, the inversion is more accurate. We can clearly see the two wings, the two floats and the fin of the biplane.

7 Conclusions

In this paper, we have presented FaIMS, a method for the inverse medium problem for the time-harmonic scalar wave equation. FaIMS uses a randomized SVD algorithm to compute SVDs of small submatrices and then applies a recursive SVD algorithm to reconstruct the overall factorization. Its complexity estimate is orders-of-magnitude smaller than the standard SVD factorization. The method is matrix-free, it only requires matrix-vector multiplication for the forward and adjoint problems. We showed that the factorization error in the singular values is bounded by the smallest largest singular value that we truncate in the rank approximation. The numerical efficiency and accuracy of the method is demonstrated in several numerical experiments in the low frequency (0-10 wavelengths) regime for the case of point scatterers. FaIMS can handle detectors and sources located on arbitrary geometries.

In future work, we intend using our approximate SVD factorization as a preconditioner with a Newton-Krylov-Multigrid iterative method for full nonlinear inversion method (for example, for problems in which G is not analytically available [5]). Another interesting question is how do tune the accuracy of the approximation to $\mathbf{M}$ given a noise level and a regularization operator. Also, our ongoing work includes adaptive algorithms and parallelization of the method. For higher frequencies, ideas discussed in [13] can be explored to construct directional low-rank approximations.

References

[1] S. S. ADAVANI AND G. BIROS, *Fast algorithms for source identification problems with parabolic PDE constraints*, SIAM Journal on Imaging Sciences, (2010). in press.

[2] V. AKCELIK, G. BIROS, AND O. GHATTAS, *Parallel multiscale Gauss-Newton-Krylov methods for inverse wave propagation*, in Proceedings of the IEEE/ACM SC2002 Conference, The SCxy Conference series, Baltimore, Maryland, November 2002, ACM/IEEE.

[3] U. ASCHER AND E. HABER, *A multigrid method for distributed parameter estimation problems*, ETNA, 15 (2003).

[4] W. BANGERTH, *A framework for the adaptive finite element solution of large-scale inverse problems*, SIAM Journal on Scientific Computing, 30 (2008), pp. 2965–2989.

[5] G. BIROS AND G. DOĞAN, *A multilevel algorithm for inverse problems with elliptic PDE constraints*, Inverse Problems, 24 (2008), pp. 1–19.

[6] A. BORZÍ, *High-order discretization and multigrid solution of elliptic nonlinear constrained optimal control problems*, Journal Of Computational And Applied Mathematics, 200 (2007), pp. 67–85.

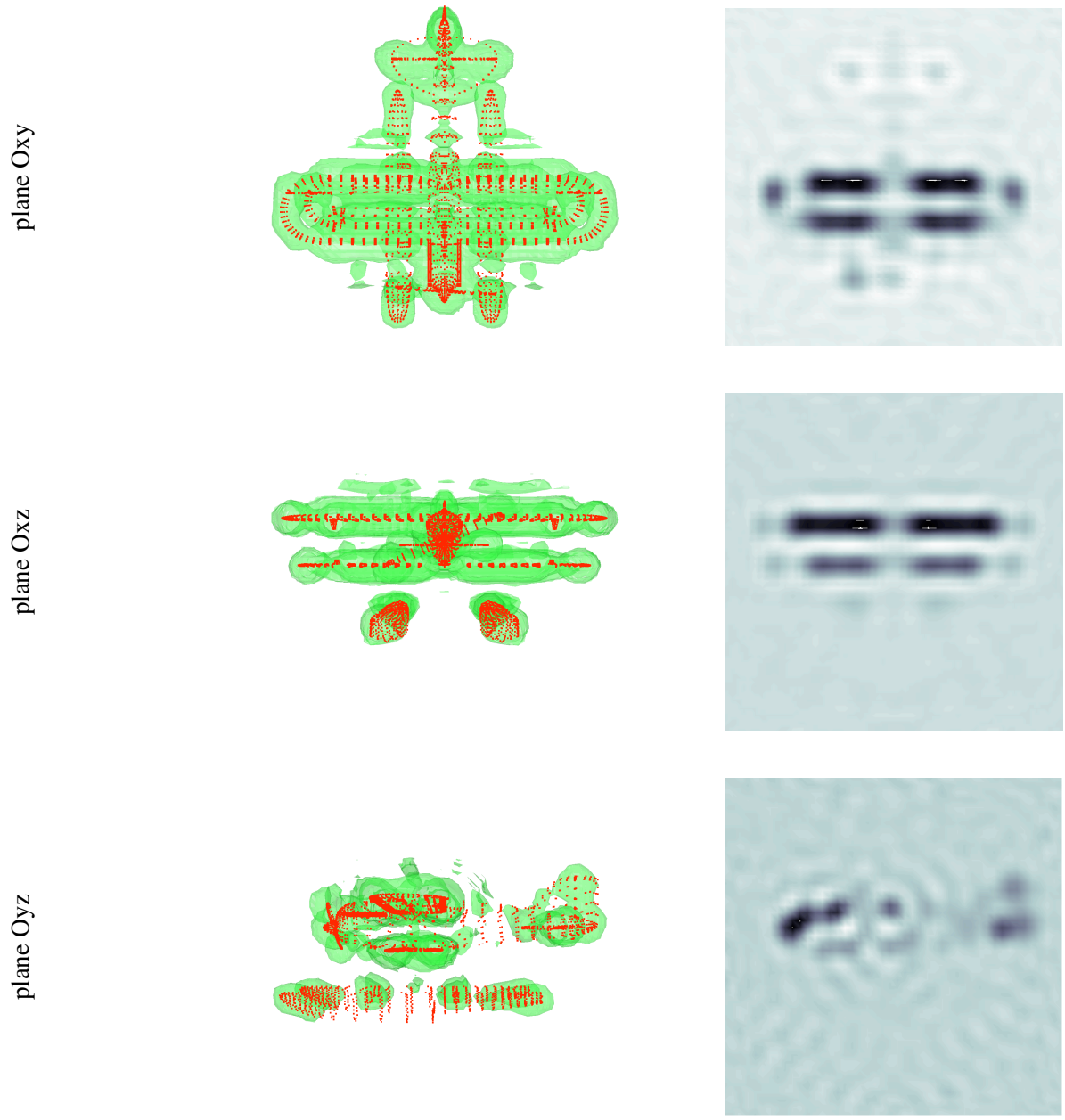


Figure 12: *Results of the inversion for the scatterer model of the biplane with size* 5λ*. Each row corresponds to a particular point of view (plane Oxy, Oxz or Oyz). The left column corresponds to a 3-D view of the isosurfaces* ($\eta = 0.25 * \eta_{max}$) *and the right column corresponds to a 2-D slice, respectively from top to bottom at the planes* $z = 0$*,* $y = 0$ *and* $x = 0$*.*

[7] E. CANDÈS AND J. ROMBERG, *Sparsity and incoherence in compressive sampling*, Inverse Problems, 23 (2007), pp. 969–985.

[8] H. CHENG, L. GREENGARD, AND V. ROKHLIN, *A fast adaptive multipole algorithm in three dimensions*, Journal of Computational Physics, 155 (1999), pp. 468–498.

[9] J. CHUNG, J. NAGY, AND D. OŠLEARY, *A Weighted-GCV Method for Lanczos-Hybrid Regularization*, Electronic Transactions on Numerical Analysis, 28 (2008), pp. 149–167.

[10] D. COLTON AND R. KRESS, *Inverse Acoustic and Electromagnetic Scattering Theory, 2nd Edition*, Applied Mathematical Sciences, Springer, 1998.

[11] J. DEMMEL, L. GRIGORI, M. HOEMMEN, AND J. LANGOU, *Communication-optimal parallel and sequential QR and LU factorizations*, Arxiv preprint arXiv:0808.2664, (2008).

[12] E. ELMROTH AND F. GUSTAVSON, *Applying recursion to serial and parallel QR factorization leads to better performance*, IBM Journal of Research and Development, 44 (2000), pp. 605–624.

[13] B. ENGQUIST AND L. YING, *Fast directional multilevel algorithms for oscillatory kernels*, SIAM Journal on Scientific Computing, 29 (2008), pp. 1710–1737.

[14] H. FLATH, L. WILCOX, V. AKÇELIK, J. HILL, B. VAN BLOEMEN WAANDERS, AND O. GHATTAS, *Fast algorithms for bayesian uncertainty quantification in large-scale linear inverse problems based on low-rank partial hessian approximations*, SIAM Journal on Scientific Computing, 33 (2011), pp. 407–432.

[15] G. H. GOLUB AND C. H. V. LOAN, *Matrix Computations*, Johns Hopkins, third ed., 1996.

[16] E. HABER AND U. ASCHER, *Preconditioned all-at-one methods for large, sparse parameter estimation problems*, Inverse Problems, 17 (2001), pp. 1847–1864.

[17] E. HABER, S. HELDMANN, AND U. ASCHER, *Adaptive finite volume method for distributed non-smooth parameter identification*, Inverse Problems, 23 (2007), pp. 1659–1676.

[18] N. HALKO, P. MARTINSSON, AND J. TROPP, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Review, 53 (2011), p. 217.

[19] F. HERRMANN, Y. ERLANGGA, AND T. LIN, *Compressive simultaneous full-waveform simulation*, Geophysics, 74 (2009), p. A35.

[20] T. HOHAGE, *On the numerical solution of a three-dimensional inverse medium scattering problem*, Inverse Problems, 17 (2001), p. 1743.

[21] T. KOLDA AND B. BADER, *Tensor decompositions and applications*, SIAM review, 51 (2009).

[22] J. KREBS, J. ANDERSON, D. HINKLEY, R. NEELAMANI, S. LEE, A. BAUMSTEIN, AND M. LACASSE, *Fast full-wavefield seismic inversion using encoded sources*, (2009).

[23] E. LIBERTY, F. WOOLFE, P. MARTINSSON, V. ROKHLIN, AND M. TYGERT, *Randomized algorithms for the low-rank approximation of matrices*, Proceedings of the National Academy of Sciences, 104 (2007), p. 20167.

[24] M. MAHONEY AND P. DRINEAS, *Cur matrix decompositions for improved data analysis*, Proceedings of the National Academy of Sciences, 106 (2009), p. 697.

[25] V. A. MARKEL, V. MITAL, AND J. C. SCHOTLAND, *Inverse problem in optical diffusion tomography. iii. inversion formulas and singular-value decomposition*, Journal of the Optical Society of America A, 20 (2003), pp. 890–902.

[26] P. MARTINSSON, V. ROKHLIN, AND M. TYGERT, *A randomized algorithm for the decomposition of matrices*, Applied and Computational Harmonic Analysis, (2010).

[27] J. L. MORALES AND J. NOCEDAL, *Automatic preconditioning by limited memory quasi-Newton updating*, SIAM Journal on Optimization, 10 (2000), pp. 1079–1096.

[28] R. NEELAMANI, C. KROHN, J. KREBS, M. DEFFENBAUGH, J. ANDERSON, AND J. ROMBERG, *Efficient seismic forward modeling using simultaneous random sources and sparsity*, submitted to Geophysics, (2009).

[29] C. PAIGE AND M. SAUNDERS, *Algorithm 583: LSQR: Sparse linear equations and least squares problems*, ACM Transactions on Mathematical Software (TOMS), 8 (1982), pp. 195–209.

[30] G. STEWART, *On the perturbation of pseudo-inverses, projections and linear least squares problems*, SIAM review, 19 (1977), pp. 634–662.

[31] ——, *Perturbation theory for the singular value decomposition*, Computer Science Technical Report Series; Vol. CS-TR-2539, (1990), p. 13.

[32] A. TARANTOLA, *Inverse problem theory and methods for model parameter estimation*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2005.

[33] L. YING, G. BIROS, AND D. ZORIN, *A kernel-independent adaptive fast multipole method in two and three dimensions*, Journal of Computational Physics, 196 (2004), pp. 591–626.